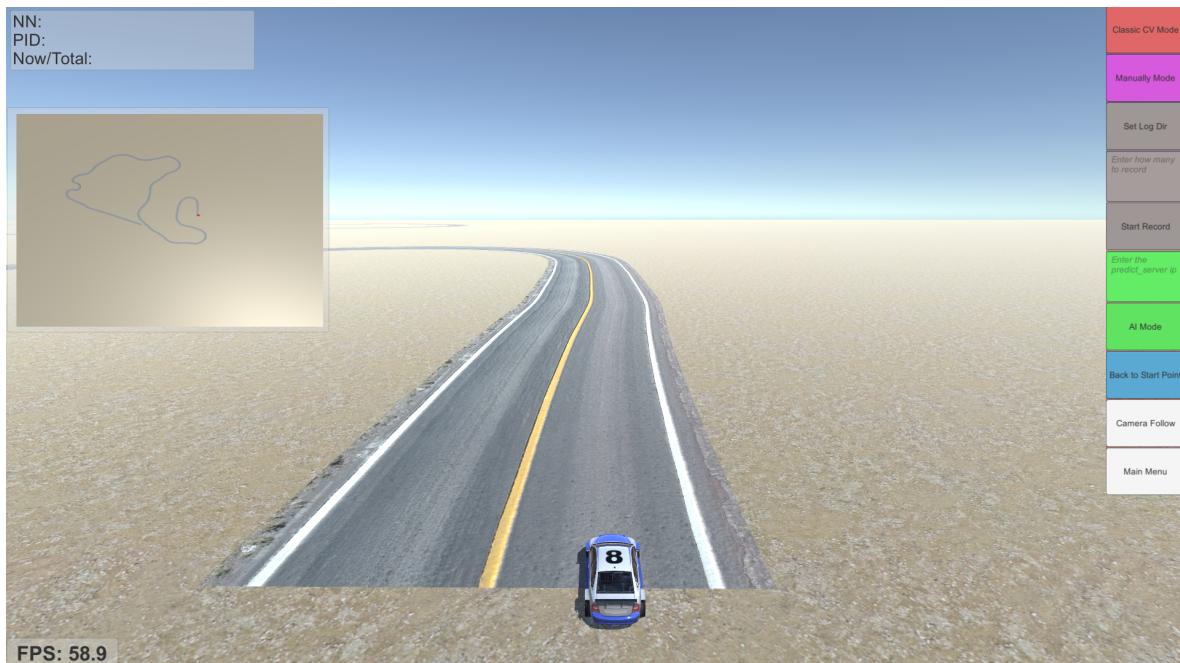


DPU+Vitis-AI+Unity3D Simulator

<!--

- @Author: Wenjie Wang
- @Email: xup_china.xilinx.com
- @Date: 2020-11-06

-->



简介

本项目融合了DPU、Vitis-AI和Unity3D游戏模拟器，让想要学习DPU和Vitis-AI的开发人员通过玩游戏的方式一步步熟悉Vitis-AI工具链的使用流程以及DPU应用的部署步骤，而无需搭建额外的硬件试验环境，避免了硬件带来的诸多不确定性，同时也降低了开发者学习的难度，使DPU和Vitis-AI的学习变得更加有趣。

1. 游戏模拟器

游戏模拟器是使用C#编程语言和Unity3D游戏引擎搭建的，其目的是构建以小车为主体的DPU和Vitis-AI学习实验环境。在游戏模拟器中以游戏关卡的形式将DPU和Vitis-AI的学习内容划分为具备不同难度等级的游戏任务，只有当完成了当前游戏任务的内容才能开启下一关卡的游戏。各个游戏关卡由易到难，让开发者逐步学习和了解DPU、Vitis-AI的部署开发流程。

1.1 登陆界面

刚打开游戏模拟器会进入一个登陆界面（欢迎界面）如下：



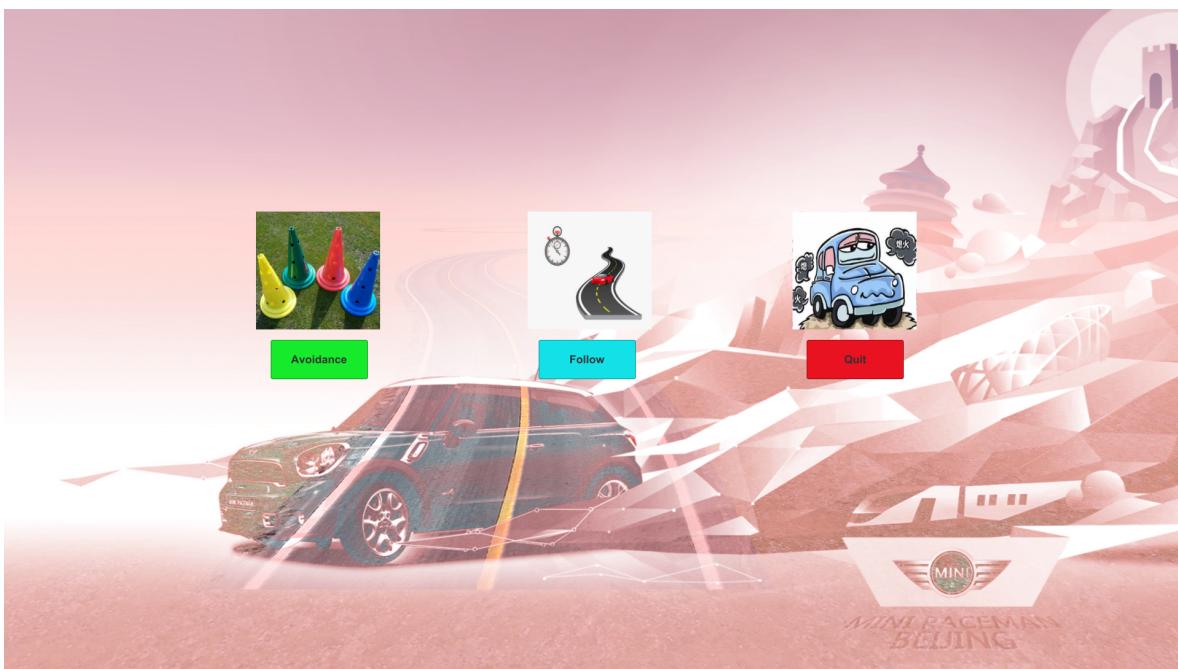
首次登录需要输入一个用户名和密码，后台数据库会将关卡解锁情况和游戏得分信息保存至该用户目录下。

此外，登录界面还有两个功能按钮如下：

- Start：进入游戏关卡选择界面（若用户名和密码任意一个为空，则弹出提示信息，反之进入游戏关卡选择界面）；
- Quit：退出游戏。

1.2 游戏关卡选择界面

游戏关卡选择界面如下：



游戏关卡选择界面主要实现不同游戏任务的选择功能，并附带有关卡锁定和解锁的功能（尚在开发），即依据登录界面输入的用户信息得出当前用户哪些关卡已经通关（解锁），哪些关卡尚未通关（锁定）。默认第一关卡是解锁的，剩余其他关卡需要通过完成上一关卡任务来解锁。暂定有如下6个游戏关卡，由易到难来一步步带领开发者熟悉DPU和Vitis-AI的整个开发流程。

- 游戏关卡1：跟踪（预先提供yolo模型）；一尚在开发
- 游戏关卡2：避障（预先提供CNN模型）；一已经完成开发

- 游戏关卡3：巡线+避障（预先提供CNN模型）；一已经完成开发
- 游戏关卡4：红绿灯检测（开发者自行搭建算法模型，并完成Vitis-AI的编译和DPU的部署）；一尚在开发
- 游戏关卡5：路标识别（开发者自行搭建算法模型，并完成Vitis-AI的编译和DPU的部署）；一尚在开发
- 游戏关卡6：巡线+路标识别+红绿灯检测（复杂场景）（开发者自行搭建算法模型，并完成Vitis-AI的编译和DPU的部署）。一尚在开发

上述游戏关卡由易到难，下一关卡的解锁条件是上一关卡的顺利通过，每个关卡的完成情况（得分情况）均会在当前用户目录下进行实时记录（该功能尚在开发）；点击游戏关卡图标下方的按钮即可进入相应的游戏任务（前提条件是当前游戏任务已经解锁）。

1.3 游戏关卡示例一避障

接下去笔者将以避障这个游戏关卡为例来介绍一个游戏任务中的基本功能和使用方式。

避障游戏任务的主界面如下：



通过观察该界面可以发现界面的右侧有一系列功能按钮和信息输入框，它们的详细情况如下：

- ManuallyDrive：手动驾驶小车模式，是图像训练集收集时的小车控制方式；
- AI Mode：AI驾驶小车模式，是深度神经网络算法模型训练完成并完成DPU部署时的AI自动驾驶模式；
- Enter the predict_server ip：搭载了DPU的AI加速推理平台（如ZCU104）通过TCP/IP的方式与运行在PC机上的游戏模拟器交互，该信息输入框用于告知游戏模拟器DPU平台的IP地址以建立TCP/IP连接，端口默认9091；
- BuildRoad：构建避障关卡中的游戏道路环境；
- SetLogDir：选择训练集图像的保存路径；
- Enter how many to record：输入训练集图像收集的数量；
- StartRecord：点击一次开始训练集图像的收集，再点击一次暂停训练图像的收集；
- CameraFollow：设置摄像头跟随模式，点击一次主摄像头跟随小车一起运动（建议在此模式下完成游戏任务，方便小车的控制），再点击一次主摄像头固定在一点不动；
- ReturnMode：返回游戏关卡选择界面。

此外，游戏关卡界面的左上角还有一个信息实时显示栏，其主要内容如下：

- NN：在AI自动驾驶小车的模式下实时显示DPU神经网络预测的小车速度和转向数据；

- PID：在收集训练集的时候部分游戏关卡可选择PID自动控制小车运行的方式，在此方式下实时显示PID控制算法拟合的小车的实时速度和转向数据；
- Now/Total：在开始收集训练集图像时实时显示当前已收集图像的数量和需要收集图像的总数量等信息。

2.DPU AI 加速推理平台

DPU AI 加速推理平台指的是搭载了DPU的云/边缘端Xilinx深度学习加速板卡，例如U50/ZCU104。游戏模拟器通过游戏关卡中位于小车前部的摄像头实时采集模拟环境中的道路图像信息，并将每一帧图像通过TCP/IP发送至U50/ZCU104，随后U50/ZCU104使用DPU对接收到的图像进行深度神经网络加速推理，预测出小车的速度和转向数据并将其通过TCP/IP发送至PC端上的游戏模拟器，最后游戏模拟器通过接收到的小车速度和转向数据控制小车完成游戏关卡中的自动驾驶任务。

2.1 DPU AI 加速推理平台所需开发环境

(1) PYNQ开发环境

DPU AI 加速推理平台主要完成的任务是通过TCP/IP接收来自游戏模拟器的图像，并对图像进行DPU深度神经网络推理，最后将推理结果再次通过TCP/IP发送至PC端上的游戏模拟器。此间涉及到TCP/IP通信协议处理、图像数据预处理、预测结果数据封装等等，需要开发者具备相关应用的编程知识，为降低开发者的使用和编程难度，笔者推荐使用Xilinx的统一软件架构—PYNQ，使用PYNQ还有如下好处：

- 安装方便；
- 模拟器和加速平台交互便捷；
- 编程语言统一（均为Python）；
- 便于二次开发。

(2) DPU-PYNQ开发环境

在加速平台上完成深度神经网络加速推理的处理器是DPU，因此为降低DPU应用的部署难度，笔者建议在加速平台上部署DPU-PYNQ开发环境，通过Xilinx提供的Python API快速、便捷地完成基于DPU的应用部署。

2.2 完成DPU部署所需的PC开发环境

(1) 深度学习训练环境

- python；
- tensorflow/caffe/pytorch；
- numpy；
- opencv；
- matplotlib；
- sklearn等；

上述环境用于深度神经网络的训练。

(2) Vitis-AI开发环境

1. Vitis AI的官方安装教程见：<https://github.com/Xilinx/Vitis-AI>；
2. 参考在虚拟机下安装vitis-ai的教程：[./doc/1. vitis AI的安装.md](#)。

上述环境用于将tensorflow/caffe/pytorch等AI框架生成的深度神经网络模型经Vitis-AI工具链转换、量化、编译和优化后生成可部署在DPU上的.elf可执行文件，完成基于DPU的深度神经网络应用部署。

3.应用示例
